VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EDITOR MATEMATICKÝCH ROVNIC PRO ANDROID

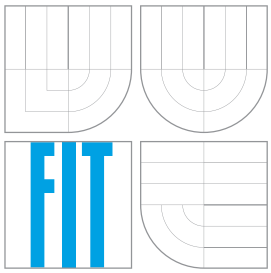
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ADAM JANIČKO

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EDITOR MATEMATICKÝCH ROVNIC PRO ANDROID

SOLVER OF MATHEMATICAL EQUATIONS FOR ANDROID

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ADAM JANIČKO

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL ZACHARIÁŠ

BRNO 2015

Abstrakt

Tato bakalářská práce se zabývá návrhem a implementací editoru rovnic pro platformu Android. Součástí aplikace je grafické uživatelské rozhraní (GUI), možnost ukládat, editovat rovnice, výpočet rovnic a widget pro rychlejší vyhodnocování již uložených rovnic. V rovnicích je uživatel schopen používat kromě vybraných matematických funkcí a konstant i vlastní proměnné, jejichž hodnotu si je schopen nadefinovat při vyhodnocování rovnice. Rovnice ve standardní infixové notaci, nevhodné pro další zpracování je převedena na prefixovou notaci upraveným Shunting Yard algoritmem. Samotný výraz, který je vyhodnocen se sestaví jednou, což znamená rychlejší vyhodnocení v případě změny hodnot uživatelem definovaných proměnných.

Abstract

This thesis aims at creating equation solver application for Android platform. The thesis includes graphical user interface (GUI), save and edit options for equations, the actual equation solving and widget for quickly evaluating already stored equations. When creating equations, user is able to use selected mathematical functions, constants and custom variables that user can define right before evaluation. The equation in the standard infix notation inappropriate for further processing is transferred to the prefix notation using modified Shunting Yard algorithm. Final expression is put together once. That means the evaluation is faster, when the values of user-defined variables changes.

Klíčová slova

Android OS, editor rovnic, Java, Shunting Yard, infixová, prefixová, postfixová notácia, widget, Android Studio, GUI.

Keywords

Android OS, equation solver, Java, Shunting Yard, infix, prefix, postfix notation, widget, Android Studio, GUI.

Citace

Adam Janičko: Editor matematických rovnic pro Android, bakalářská práce, Brno, FIT VUT v Brně, 2015

Editor matematických rovnic pro Android

Prohlášení

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Michala Zachariáša. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

. . . . . .
Adam Janičko
19. května 2015

Obsah

Kapitola 1

Úvod

Rozmach inteligentných mobilných zariadení otvoril nový trh takzvaných mobilných aplikácií. Tento trh sa od svojho vzniku neustále rozrastá a existuje už mnoho aplikácií, ktoré vykonávajú veľké množstvo rôznych činností od zábavy po zjednodušovanie bežných úkonov všedného dňa, či už v práci alebo doma. Preto som sa rozhodol využiť túto bakalársku prácu ako príležitosť na zapojenie sa do tohto spoločenstva vývojárov mobilných aplikácií a vytvoriť aplikáciu, ktorá by užívateľom priniesla niečo nové alebo lepšie než ponúkajú už existujúce aplikácie.

Aplikáciu, ktorú som vytvoril ja, a ktorú popisuje táto technická správa, má užívateľom priniesť určitú voľnosť pri tvorbe vlastných matematických rovníc, ktorá chýba v už aktuálne existujúcich aplikáciách.

Tak ako počítače, aj mobilné zariadenia majú rôznych výrobcov, hardware a operačné systémy. Momentálne sa aplikácie vyvíjajú pre tri systémy a to Android OS, iOS a Windows Phone. Každý operačný systém má svoje plusy a mínusy, no jeden z hlavných dôvodov prečo som sa rozhodol pre Android OS, je podiel jeho zariadení na trhu a fakt, že som vlastníkom zariadenia s týmto operačným systémom, čo mi vo výsledku uľahčilo testovanie na reálnom zariadení.

V druhej kapitole sa budem venovať teoretickým poznatkom, ktoré sú potrebné pri tvorbe mobilných aplikácií. Predstavím operačný systém Android, priblížim nástroje, vývojové prostredia a postupy, ktoré by sa pri písaní aplikácií mali dodržiavať. Ďalej taktiež poukážem na problémy tvorby užívateľského rozhrania pre mobilné zariadenia a ich najbežnejšie riešenia 2.

Tretia kapitola sa zaoberá rozborom zadania práce a návrhu jednotlivých komponentov výslednej aplikácie. Medzi tieto komponenty patrí napríklad návrh užívateľského rozhrania, modelu ukladania rovníc a ich vyhodnocovania 3.

Vo štvrtej kapitole je popísaná samotná realizácia a taktiež testy zamerané na správnosť návrhu užívateľského rozhrania, a funkčnosti aplikácie 4.

Kapitola 2

Teória

Táto kapitola je venovaná teoretickým poznatkom, ktoré som nadobudol pri riešení tejto bakalárskej práce. Týkajú sa vývoja aplikácií pre platformu Android, ktorý prebieha v programovacom jazyku Java. Vývoj zahŕňa implementáciu moderného užívateľského rozhrania a funkčnosti za ním. Taktiež by som chcel poukázať na už existujúce podobné aplikácie a odlišnosti v nich oproti mojej aplikácií.

2.1 Android OS

Výtvor Andyho Rubin-a nazvaný Android bol v roku 2005 kúpený spoločnosťou Google a voľne sprístupnený vývojárom na tvorbu aplikácií pre mobilné zariadenia prostredníctvom programovacieho jazyku Java a XML. Odvtedy fenomén s názvom Android umožnil vznik aliancie popredných výrobcov (Samsung, . . .), ktorá sa stala najrýchlejšie rastúcou mobilnou platformou dneška.

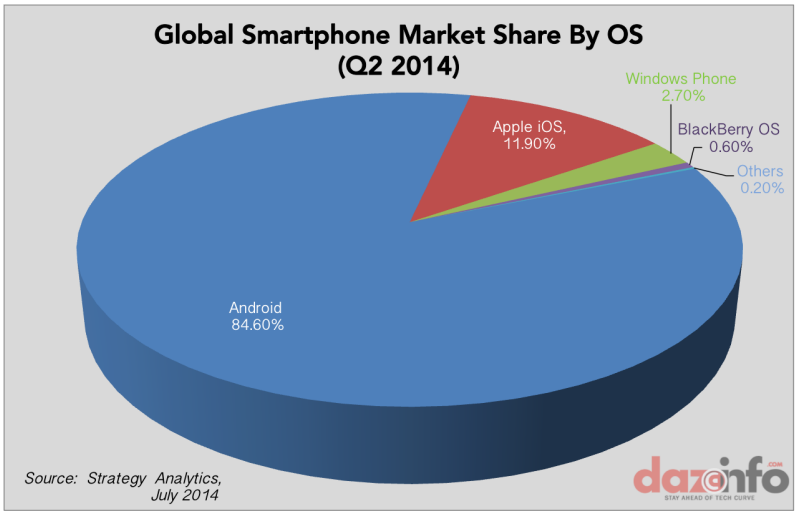
Android OS je operačný systém založený na Linuxe, ktorého podiel na trhu bol minulý rok 84,6% (viď obrázok 2.1). Android OS je licencovaný ako open source projekt pod licenciou Apache 2.0 [7].

Android je navrhnutý tak, aby bežal na rôznych zariadeniach, od telefónov cez tablety až po televízie. Pre vývojárov, veľké spektrum zariadení predstavuje množstvo potencionálnych používateľov ich aplikácií. Aby boli aplikácie úspešné na všetkých týchto zariadeniach s rôznymi displejmi je nutné, aby aplikácie tolerovali určitú variabilitu vo funkčnosti, a poskytovali flexibilné užívateľské rozhrania, ktoré sa adaptujú na rôzne konfigurácie obrazoviek týchto zariadení.

Android je operačný systém, v ktorom každá aplikácia beží s odlišnou systémovou identitou. Časti operačného systému sú taktiež rozdelené do odlišných identít. Táto funkčnosť umožňuje Linux-u izolovať aplikácie jednu od druhej a od operačného systému.

Nástroje Android SDK (Software Developer Kit) kompilujú kód aplikácie spolu s dátami a zdrojovými súbormi do APK. APK alebo *Android package*, je archívny súbor s príponou *.apk*. APK súbor obsahuje všetok obsah Android aplikácie a je to súbor, ktorý používajú zariadenia so systémom Android na inštaláciu aplikácie [17].

Hlavným prvkom každej Android aplikácie sú *Activities*. Aktivita reprezentuje základnú vizuálnu komponentu, ktorá zobrazuje jednu „obrazovku“ aplikácie. Vzhľad aktivity je definovaný pomocou súborov XML. Všetky akcie (stlačenie tlačidla, pridanie znaku z klávesnice, pohyb prstom po displeji, . . .), ktoré užívateľ vykoná nad danou obrazovkou sú spracované príslušnou aktivitou, ktorá obrazovku vytvorila. Aktivita môže byť v rozličných stavoch,



Obrázek 2.1: Smartphone Market Share 2014 [20]

ktoré závisia na tom ako užívateľ s aktivitou pracuje. Každý stav má metódu, ktorá sa zavolá v momente keď daný stav nastane (viď obrázok 2.2) [24].

Testovanie aplikácie je možné na emulátore alebo reálnom prístroji. Emulátor pracuje pomalšie, ale poskytuje vývojárom možnosti nastavenia veľkosti, rozlíšenia displeja alebo verzie Android OS, ktorá má bežať na emulovanom zariadení. Existuje viacero emulátorov, ktoré môžeme použiť pri vývoji na testovanie našej aplikácie. Vývojové prostredie Android Studio obsahuje vlastný vstavaný emulátor, ktorý je použiteľný no pomalý. Existujú emulátory tretej strany ako napríklad Genymotion emulátor (viď obrázok 2.3), ktorý som použil pri vývoji aplikácie, ktorú popisuje táto bakalárska práca. Tento emulátor je nová generácia AndroVM open source projektu, ktorý používa už viac ako 2,500,000 vývojárov. Jeho hlavné výhody oproti emulátoru, ktorý je vstavaný do Android Studia je jeho rýchlosť a jednoduchosť.

Pri vývoji aplikácií vývojár používa viacero vývojových nástrojov.

Java Development Kit (JDK) – vývojová sada, ktorá obsahuje nástroje a knižnice Javy.

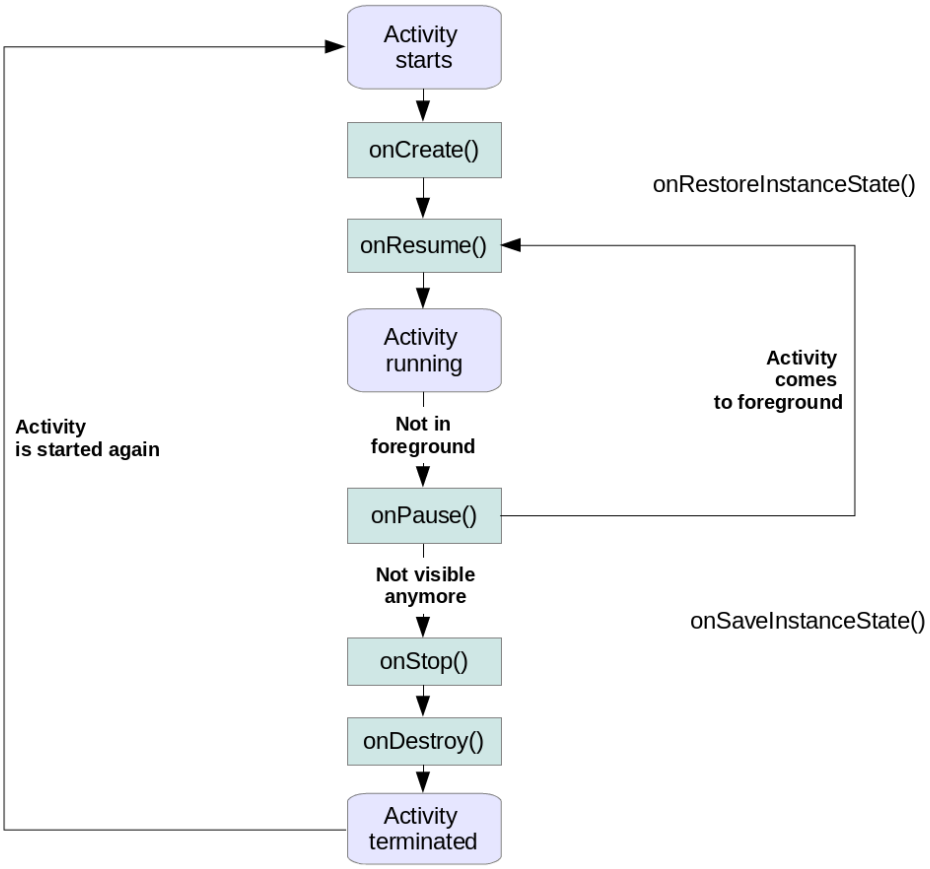
Android Software Development Kit(SDK) – vývojová sada, ktorá umožňuje tvorbu aplikácií pre platformu Android. Obsahuje vzorové projekty, nástroje, emulátor a knižnice potrebné pre vývoj aplikácií.

Android Studio/Eclipse – vývojové prostredia pre vývoj na Android platforme.

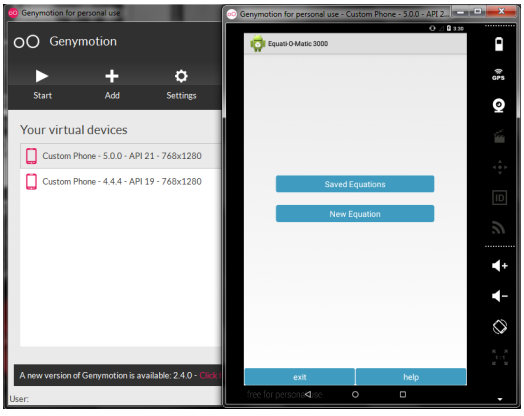
Táto bakalárska práca bola písaná v Android Studio-u (viď obrázok 2.5). Eclipse ADT už nieje podporovaný Google-om a tak je Android Studio odporúčané a preferované vývojárskou komunitou.

2.2 Riešenie rovníc

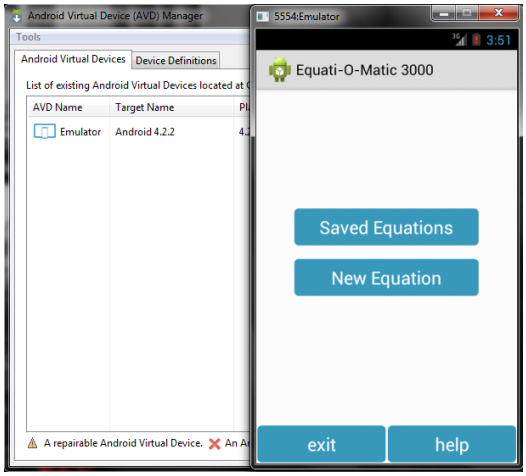
V matematike, vyriešiť rovnicu znamená nájsť hodnoty(čísla, funkcie, ...), ktoré spĺňajú podmienku stanovenú vo forme rovnice. Pri hľadaní riešenia, jedna alebo viac premenných



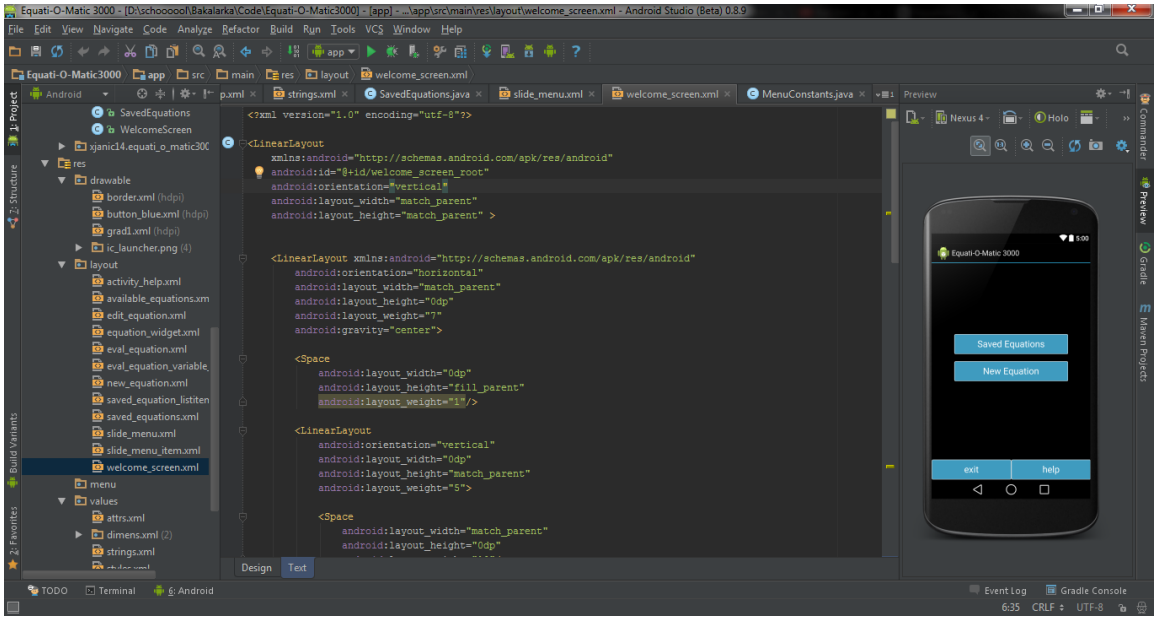
Obrázek 2.2: Activity lifecycle [24]



Obrázek 2.3: Genymotion emulátor



Obrázek 2.4: Android Studio emulátor



Obrázek 2.5: Android Studio

sú označené ako neznáme. Riešenie je priradenie výrazov neznámym premenným tak, aby platila rovnosť v rovnici. Inak povedané, riešenie je výraz alebo kolekcia výrazov (jeden pre každú neznámu), ktoré ak sú nahradené za neznáme spôsobia, že rovnica sa stane identitou. Riešenie rovníc môže byť numerické alebo symbolické. Pri numerickom riešení sú ako riešenia akceptované iba čísla (nie kombinácie premenných). Pri symbolickom riešení výrazy, ktoré obsahujú známe premenné alebo premenné neprítomné v pôvodnej rovnici, sú taktiež akceptované ako riešenia [5].

Rovnice (výrazy) sú bežne zadávané v infixovej notácií, ktorá je ľahko čitateľná pre ľudí, no ťažšie spracovateľná počítačom. Pri práci s výrazmi v infixovom tvare sa preto najskôr používa konverzia do postfixovej alebo prefixovej notácie (viď obrázok 4.9). Na tento prevod sa používajú takzvané *operator-precedence analyzátory*. Tieto syntaktické analyzátory sú *bottom-up, shift-reduce*, čo znamená, že najskôr identifikujú a spracujú základné jednotky(listy derivačného stromu) začínajúc na ľavej strane a z nich následne odvodzujú štruktúru s vyššou úrovňou usporiadania (viď obrázok 2.6) [6].

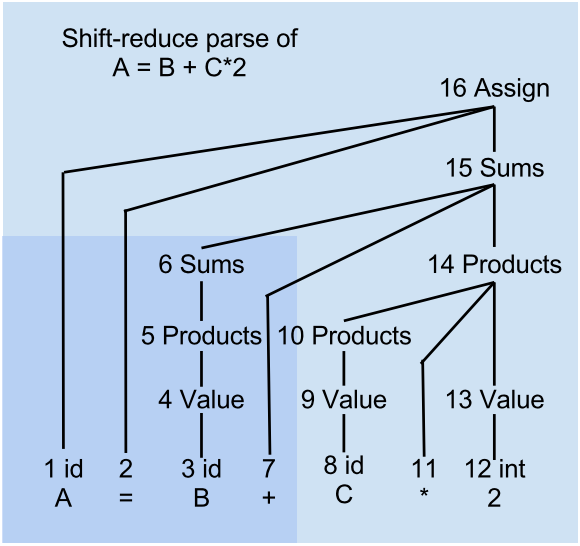
Operator-precedence analyzátory sa bežne v praxi nepoužívajú. Majú však zopár vlastností, vďaka ktorým sú v určitých prípadoch užitočné:

1. Sú dosť jednoduché na to, aby sa dali napísať ručne (bez použitia generátora analyzátorov), čo väčšinou nieje možné pri viac sofistikovaných *right shift-reduce* analyzátoroch.

2. Môžu byť napísané tak, aby dokázali pracovať s tabuľkou operátorov počas behu programu. Táto vlastnosť je výhodná pri jazykoch, ktoré môžu pridať alebo zmeniť svoje operátory počas analýzy.

Na implementáciu operator precedence analyzátorov sa používa viacero algoritmov:

- Edsger Dijkstra-ov shunting yard algoritmus
- Precedence climbing metóda
- Top down operator precedence metóda



Obrázek 2.6: Kroky syntaktickej analýzy zdola nahor

Shunting yard

Jedna sa o jeden z najjednoduchších, no v praxi nie tak často používaný algoritmus (viď obrázok 2.7) [21].

Precedence climbing

Pri tejto metóde sa výrazy skladajú z atómov, ktoré sú pospájané binárnymi operátormi. Atóm je buď číslo alebo výraz v zátvorkách. Algoritmus je vedený operátormi. Základný krok algoritmu spočíva v spracovaní ďalšieho atómu a preskúmaní operátora za ním. Ak je prednosť operátora nižšia ako najnižšia povolená hodnota, algoritmus končí. Ak nie, algoritmus je volaný v cykle z dôvodu spracovania pod výrazov [9].

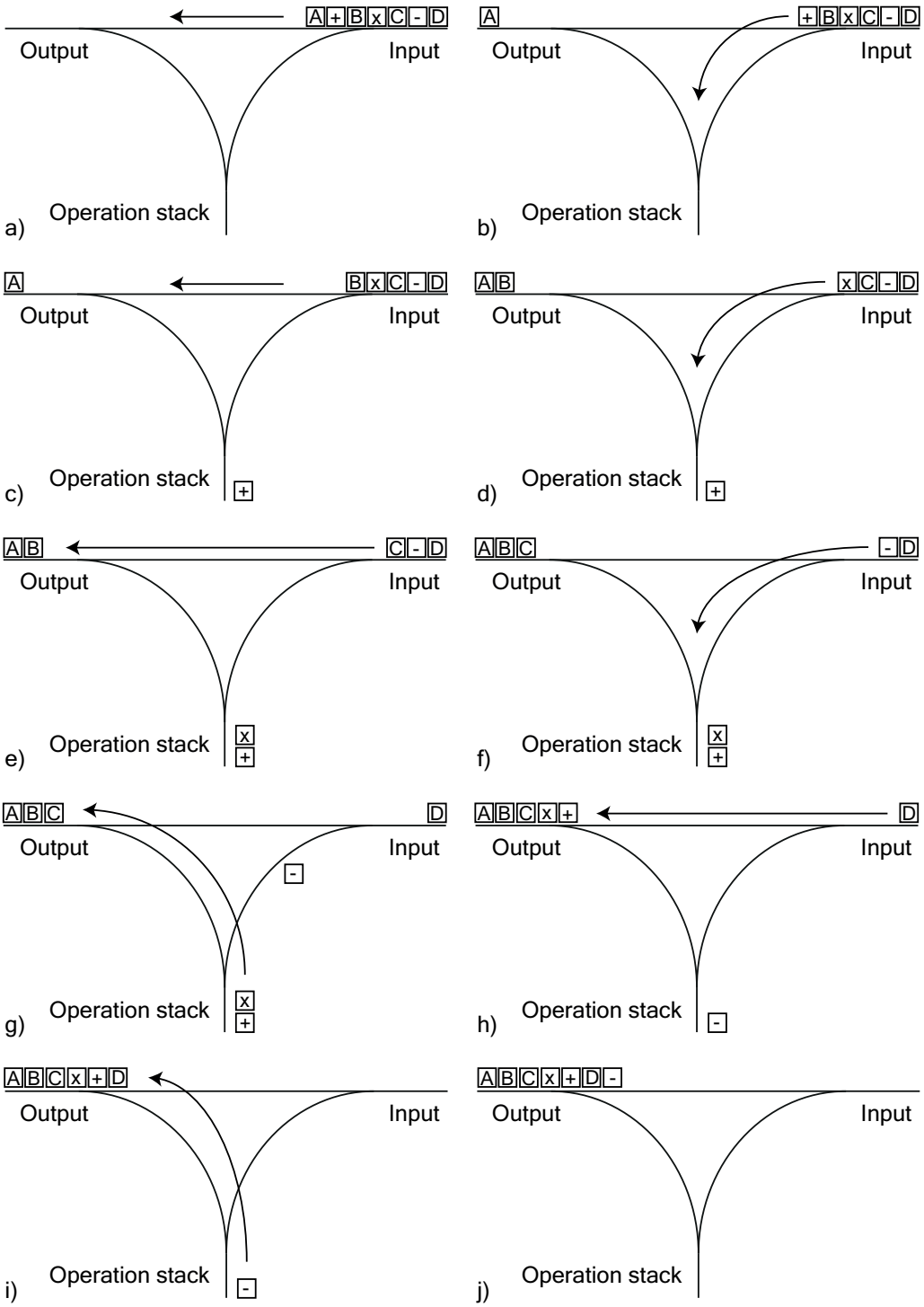
Top down operator precedence (TDOP)

Základné princípy TDOP:

- Mechanizmus „sily (Binding power)“, ktorý sa používa pri riešení prednosti operátorov.
- Prostriedky na implementovanie odlišných funkcionalít spracovávaných tokenov, v závislosti na ich pozícii k susedom (infix, prefix).
- Sémantické akcie sú spájané s tokenmy.

Ako sme si už povedali TDOP rieši prioritu operátorov implementáciou sily, ktorú priradí každému spracovávanému tokenu. Ako príklad si môžme uviesť 1+2*4. Operátor plus bude mať napríklad silu 10 a operátor krát 20. Tým, že operátor krát má väčšiu silu ako plus, číslo 2 bude „zviazané“ s operátorom krát.

Algoritmus taktiež musí pri spracovaní infixových výrazov odlišovať medzi prefixovou a infixovou formou tokenov. Príkladom môže byť operátor mínus. Mínus v infixovej forme je odpočítanie pravého operandu od ľavého. V prefixovej je to zase negácia pravého operandu [8].



Obrázek 2.7: Grafické znázornenie algoritmu Shunting Yard použitím troj-smernej železničnej križovatky. Vstup sa spracováva po symboloch. Ak je nájdená premenná alebo číslo, uloží sa priamo na výstup (a),c),e),h)). Ak je symbol operátor, uloží sa na zásobník (b),d),f)). Ak je priorita operátora menšia alebo rovná ako priorita operátorov na vrchu zásobníka a operátor je asociatívny zľava, potom sú dané operátory vybrané z vrchu zásobníka a pridané do výstupu (g)) a aktuálny operátor je pridaný na zásobník. Zvyšné operátory sú nakoniec vybrané zo zásobníka a pridané na výstup (i)) [12].

2.3 Existujúce aplikácie

Aplikácie existujú na všetko a ani počítanie rovníc nieje výnimkou. V tejto časti predstavím už existujúce aplikácie a ich porovnanie s mojou aplikáciou.

Science Equation Solver

Logo $\int\pi$

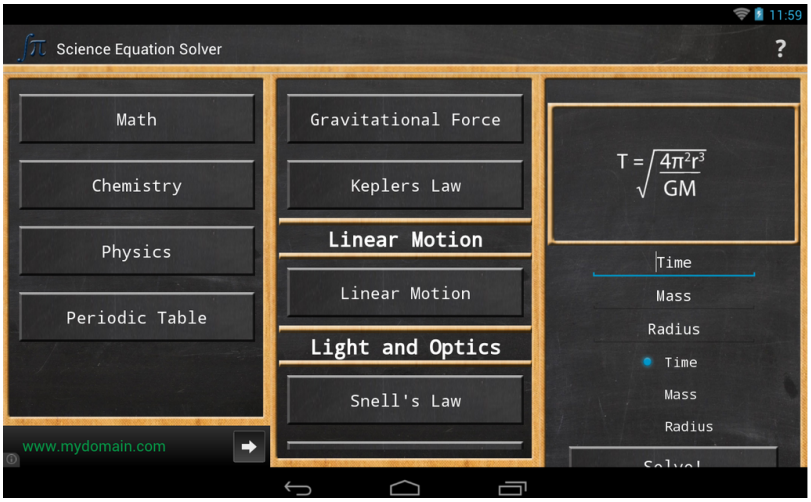
Vývojár Patrick M. Doyle

Dátum vydania August 15, 2013

Počet inštalácií 10,000 - 50,000

Táto, z časti bezplatná aplikácia, užívateľovi poskytuje možnosť počítania predpripravených rovníc z oblastí Matematiky, Chémie a Fyziky. V matematickej oblasti je užívateľ schopný riešiť kvadratické rovnice, limity a derivácie(iba platiaci užívatelia) formou dosadenia koeficientov do pripravenej šablóny. Fyzikálna oblasť poskytuje sadu známych fyzikálnych rovníc, do ktorých užívateľ dosadí za premenné požadované hodnoty a aplikácia mu spočíta výsledok. Väčšina rovníc je platená.

Aplikácia neposkytuje žiadnu možnosť pridávania vlastných rovníc, ani tvorbu widget-ov nad jednotlivými uloženými rovnicami pre rýchlejšie vyhodnocovanie bez nutnosti spúšťania aplikácie.



Obrázek 2.8: Science Equation Solver [13]

Kalkulon

Logo 

Vývojár J. Holetzeck

Dátum vydania June 8, 2013

Počet inštalácií 500 - 1,000

Kalkulon je bezplatná programovateľná výrazová kalkulačka pre programátorov. Aplikácia pozostáva z priestoru, do ktorého vkladáme príkazy ako napríklad priradenie do premennej, ktorú môžeme neskôr použiť v nasledujúcich príkazoch(rovniciach). Aplikácia podporuje množstvo vstavaných funkcií, od matematických po programátorské.

Aplikácia neposkytuje žiadnu možnosť ukladania vytvorených rovníc a keďže neexistujú žiadne uložené rovnice prítomnosť widget-u je nepotrebná.

Math Solver

Logo 

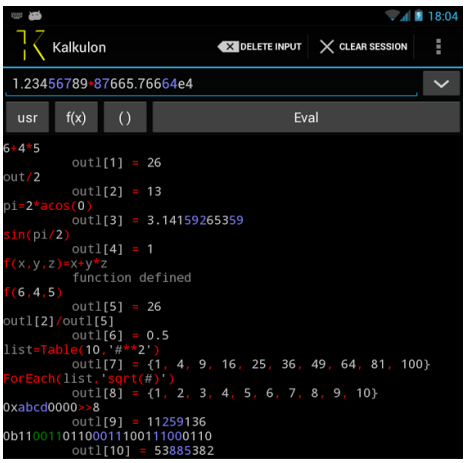
Vývojár Shakti Malik

Dátum vydania March 11, 2015

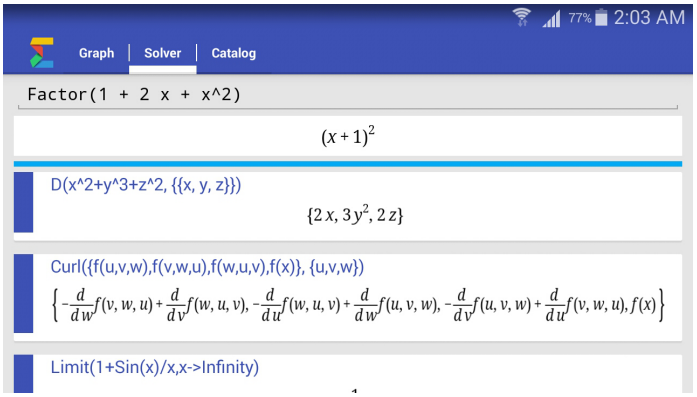
Počet inštalácií 100,000 - 500,000

Math Solver je vedecká a grafická kalkulačka s mnohými vstavanými funkciami. Rieši základné matematické problémy, algebru, komplexné čísla, trigonometriu, integrály a derivácie, množiny, matice a vektory. V neplatenej verzii obsahuje reklamy.

Aplikácia neposkytuje žiadnu možnosť ukladania vytvorených rovníc a keďže neexistujú žiadne uložené rovnice prítomnosť widget-u je nepotrebná.



Obrázek 2.9: Kalkulon [16]



Obrázek 2.10: Math Solver [19]

Programmable Calculator

Logo 

Vývojár Stanislav Burov

Dátum vydania October 14, 2012

Počet inštalácií 10,000 - 50,000

PGMCalc je programovateľná vedecká kalkulačka. Okrem klasických vedeckých funkcií disponuje schopnosťou automatizovať opakujúce sa výpočty. Táto aplikácia je implementovaná podľa Citizen SRP-145 programovateľnej kalkulačky.

Aplikácia podporuje určitú úroveň pridávania rovníc, ktoré ale pretrvávajú iba po dobu behu aplikácie a majú striktné obmedzenia pri ich tvorbe. Keďže neexistujú žiadne uložené rovnice prítomnosť widget-u je nepotrebná.

Calculate by QxMD

Logo 

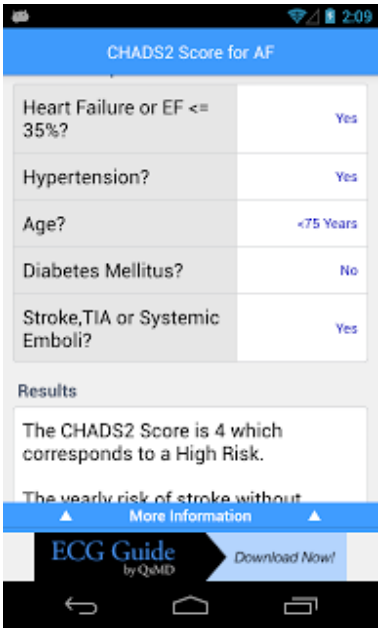
Vývojár QxMD Medical Software Inc.

Dátum vydania October 9, 2014

Počet inštalácií 500,000 - 1,000,000

Calculate je bezplatná medicínska kalkulačka a nástroj na rozhodovanie novej generácie. Obsahuje množstvo predpripravených rovníc a formulárov pre rozhodovanie do ktorých užívateľ zadáva hodnoty premenných. Aplikácia rovnicu alebo formulár s týmito hodnotami spočíta a na základe výsledku vyvodí záver.

Táto aplikácia je zaujímavá z dvoch dôvodov, ktoré by sa dali využiť pri ďalšom vývoji mojej aplikácie. Formuláre pre rozhodovanie, ktoré fungujú na základe otázok, podľa ktorých sa nastavujú premenné v rovnici a interpretácia výsledkov, podľa predom nastavených prahov. Aplikácia neposkytuje žiadnu možnosť pridávania vlastných rovníc, ani tvorbu widget-ov nad jednotlivými uloženými rovnicami pre rýchlejšie vyhodnocovanie bez nutnosti spúšťania aplikácie.



Obrázek 2.11: Calculate by QxMD [22]

Obrázek 2.12: Programmable Calculator [10]

Advanced Equation Solver

Logo 

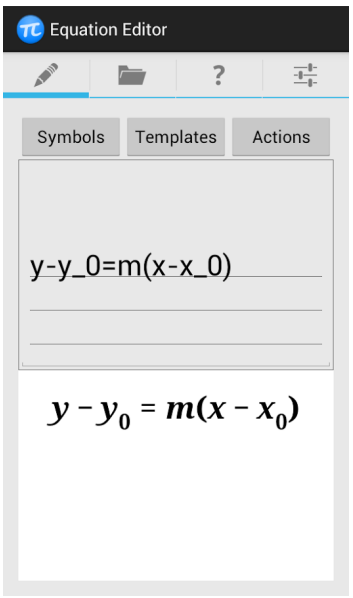
Vývojár Akira Katsukawa

Dátum vydania September 5, 2014

Počet inštalácií 500 - 1,000

Advanced Equation Solver je čiastočne bezplatný nástroj na riešenie všetkých druhov rovníc. Poskytuje špecializované vstavané funkcie. V neplatenej verzii si užívateľ môže uložiť iba štyri rovnice.

Táto aplikácia je zaujímavá z jedného dôvodu, ktorý by sa dal zapracovať do mojej aplikácie. Disponuje vlastnou na mieru prispôsobenou klávesnicou. Aplikácia nepodporuje tvorbu widget-ov z uložených rovníc.



Obrázek 2.13: Equation Editor [14]

Obrázek 2.14: Advanced Equation Solver [18]

Equation Editor

Logo

Vývojár GranadaWare

Dátum vydania July 8, 2013

Počet inštalácií 10,000 - 50,000

Equation Editor umožňuje užívateľom vytvárať a zdieľať matematické rovnice. Pre tvorbu rovníc sa používa jednoduchý popisný jazyk. Vytvorené rovnice môže užívateľ posielať svojim priateľom, zdieľať na sociálnych sieťach alebo ukladať na svoje zariadenie.

Táto aplikácia je zaujímavá z jedného dôvodu, ktorý by bol zaujímavým budúcim rozšírením do mojej aplikácie. Zdieľanie rovníc medzi priateľmi a na užívateľských sieťach. Aplikácia nepodporuje samotné vyhodnocovanie rovníc, widget je teda nepotrebný.

2.4 Užívateľské rozhranie(UI)

Užívateľské rozhranie aplikácie je všetko s čím užívateľ dokáže pracovať. Android poskytuje veľký výber predpripravených UI komponent, ako napríklad štrukturované objekty pre rozvrhnutie dizajnu a UI prvky, ktoré uľahčujú tvorbu grafického užívateľského rozhrania pre vyvíjanú aplikáciu. Android taktiež ponúka možnosť využitia ďalších UI modulov pre špeciálne rozhrania, ako napríklad dialógy, upozornenia a menu.

Všetky elementy užívateľského rozhrania v Android aplikácií sú vytvárané pomocou *View* a *ViewGroup* objektov. View je objekt ktorý vykreslí niečo na obrazovku, s čím môže

užívateľ následne pracovať. ViewGroup je objekt, ktorý obsahuje iné View (a ViewGroup) objekty za účelom definície rozloženia rozhrania.

Android poskytuje kolekciu podtried View a ViewGroup, ktoré implementujú bežné komponenty pre vstup (napr. tlačidlá a textové polia) a rôzne modely rozloženia (napr. lineárne alebo relatívne rozloženie).

Rozloženie užívateľského rozhrania

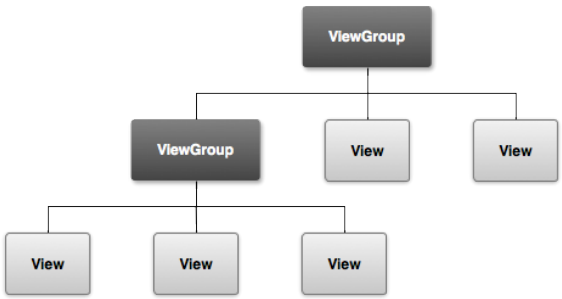
Užívateľské rozhranie pre každú komponentu aplikácie je definované pomocou hierarchie View a ViewGroup objektov (viď obrázok 2.15). Každý ViewGroup objekt je neviditeľný kontajner, ktorý organizuje potomkov, kde potomkovia môžu byť komponenty pre vstup alebo iné widget-y, ktoré vykresľujú niektoré časti UI. Tento hierarchický strom môže byť jednoduchý alebo zložitý v závislosti na potrebách aplikácie (jednoduchosť je lepšia pre výkon).

Rozloženie definuje vizuálnu štruktúru užívateľského rozhrania, ako napríklad UI pre aktivitu alebo aplikačný widget. Rozloženie môže byť deklarované dvoma spôsobmi:

- **Deklarácia UI elementov v XML** – Android poskytuje priamočiaru definíciu v jazyku XML, ktorá korešponduje s triedami a podtriedami View (napr. widget-y a rozloženia).
- **Vytváranie inštancií elementov rozloženia za behu** – aplikácia môže vytvárať View a ViewGroup objekty (a manipulovať ich nastavenia) programovo.

Android framework umožňuje používanie obidvoch týchto metód na deklarovanie a spravovanie UI aplikácie. Je napríklad možné deklarovať základné rozloženia aplikácie v XML, vrátane elementov, ktoré sa v nich zobrazia spolu s ich nastaveniami. Potom by sme v aplikácií mohli pridať kód, ktorý by modifikoval stav zobrazovaných objektov za behu aplikácie, vrátane tých, ktoré boli deklarované v XML.

Deklarovanie UI v XML umožňuje lepšie oddeliť vzhľad aplikácie od kódu, ktorý ovláda jej správanie. Popis UI je oddelený od aplikačného kódu, čo znamená, že ho môžeme modifikovať alebo adaptovať bez potreby upravovať a znovu prekladať zdrojový kód. Môžeme napríklad vytvoriť XML rozloženia pre rôzne orientácie, rozmery displejov a rôzne jazyky. Veľkou výhodou deklarovania rozloženia v XML je jednoduchšia vizualizácia štruktúry UI, čo vedie k jednoduchšiemu debugovaniu [3]. XML poskytuje štruktúru pre tvorbu rozloženia podobnú HTML, ktorá je pre ľudí viac čitateľná.



Obrázek 2.15: Ilustrácia hierarchie View a ViewGroup objektov, ktorá definuje rozloženie UI

Pri načítaní rozloženia zo zdrojového súboru XML, Android inicializuje každý uzol rozloženia do objektu vytvoreného za behu, ktorý môžeme použiť pre definíciu nadštandardného správania, dotazovania na stav objektu alebo modifikácie rozloženia.

UI nemusí byť postavené iba z View a ViewGroup objektov. Android poskytuje viacero aplikačných komponentov, ktoré ponúkajú štandardné rozloženie UI, pre ktoré potrebujeme definovať iba obsah. Každá táto UI komponenta má unikátnu sadu API (Application Program Interface), ako napríklad panel nástrojov, dialógy a systémové upozornenia [2].

Návrh dizajnu pre odlišné displeje

Android podporuje stovky typov zariadení s rôznymi veľkosťami displejov, od malých mobilných telefónov po veľké televízne obrazovky. Pri dizajne je preto dôležité, aby bol kompatibilný so všetkými veľkosťami displejov z dôvodu prístupnosti aplikácie čo najväčšiemu počtu potencionálnych používateľov.

Kompatibilita sama o sebe nestačí. Každá veľkosť displeja, ponúka odlišné možnosti a výzvy pri interakcii s užívateľom. Preto pre skutočné uspokojenie a ohúrenie užívateľov, aplikácia musí robiť viac než len *podporovať* rozličné displeje. Aplikácia musí *optimalizovať* užívateľskú skúsenosť pre konfiguráciu každého displeja [1].
Táto optimalizácia sa dá dosiahnuť dodržaním týchto zásad:

Podpora rôznych veľkostí displejov

- Zabezpečenie zmeny veľkosti rozloženia v závislosti na displeji
- Poskytnutie vhodného rozloženia UI podľa konfigurácie displeja
- Zabezpečenie použitia správneho rozloženia pre správny typ displeja
- Poskytnutie bitmáp, ktoré sa správne prispôsobujú v závislosti na displeji

Podpora rôznych hustôt pixel-ov na displeji

Pri špecifikácií rozmerov, nepoužívať absolútne pixel-y, ale dp(density-independent pixel), ktorý odpovedá fyzickej veľkosti pixel-u pri 160 dpi a sp(scale-independent pixel), ktorý prispôsobuje svoju veľkosť podľa užívateľom preferovanej veľkosti textu.

Implementácia adaptívnych tokov UI

V závislosti na aktuálnom zobrazovanom rozložení sa tok UI môže líšiť. Napríklad, ak je aplikácia v móde dvoch panelov, kliknutie na objekt v ľavom panely zobrazí obsah na pravom panely. Ak je aplikácia v móde jedného panelu, pri kliknutí na objekt by mal byť obsah zobrazený v inej aktivite.

Widget

Widget-y sú najzákladnejšie aspekty pri prispôsobovaní domácej obrazovky Android zariadenia. Sú to vlastne náhľady na najdôležitejšie dáta a funkcionalitu aplikácie prístupné z domácej obrazovky užívateľa. Užívatelia môžu widget-y presúvať a ak to daná aplikácie podporuje, meniť ich veľkosť podľa vlastných potrieb. Existuje viacero typov widget-ov:

Informačné widget-y

Typicky zobrazujú pár kritických informácií, ktoré sú dôležité pre užívateľa a sledujú ako sa tieto informácie menia v priebehu času. Ako príklad môžme uviesť widget-y s počasím, hodinami alebo športovým skóre. Pri kliknutí na informačný widget sa typicky spustí príslušná aplikácia, kde sú zobrazené detaily o informáciach na widget-e.

Zberné widget-y

Špecializujú sa v zobrazovaní množstva elementov rovnakého typu ako napríklad kolekcie obrázkov z aplikácie galéria, článkov z novinovej aplikácie alebo emailov/správ z komunikačnej aplikácie. Zberné widget-y sa typicky sústredia na dve veci: prehliadanie kolekcií a otvorenie detailu elementu kolekcie. Zberné widget-y podporujú vertikálne posúvanie.

Kontrolné widget-y

Hlavný účel kontrolných widget-ov je zobrazovanie často používaných funkcií, ktoré môže užívateľ použiť priamo z domácej obrazovky bez nutnosti otvárania aplikácie. Sú to v podstate diaľkové ovládače aplikácie. Typický príklad kontrolného widget-u je hudobná aplikácia s widget-om, ktorý umožňuje užívateľovi prehrať, prerušiť alebo preskočiť skladby mimo hudobnej aplikácie. Interakcia s kontrolnými widget-mi môže, ale nemusí vyústiť do prislúchajúceho detailného náhľadu ako napríklad v prípade widget-u určeného na hľadanie.

Hybridné widget-y

Zatiaľ čo sa všetky widget-y prikláňajú k jednému z typov popísaných vyššie, mnoho z nich sú v skutočnosti hybridy, ktoré kombinujú elementy odlišných typov. Pri plánovaní widget-u je dobré zamerať sa na jeden zo základných typov a pridať ostatné potrebné elementy.

Widget-y majú určité obmedzenia, vďaka ktorým ich nemôžeme považovať za „mini aplikácie“, a ktoré musíme pochopiť predtým než s nimi začneme pracovať.

Tým, že widget-y musia koexistovať s navigáciou stanovenou na domácej obrazovke, podpora gest je veľmi limitovaná. Jediné gestá k dispozícii sú:

- Dotyk
- Vertikálne potiahnutie

Niektoré stavebné bloky UI, ktoré sa spoliehajú na gestá sú kvôli týmto obmedzeniam pri tvorbe widget-u nedostupné [4].

Kapitola 3

Návrh

V tejto kapitole bližšie popíšem zadanie a všetky komponenty aplikácie, ktoré som implementoval. Komponenty potrebné na splnenie zadania sú: užívateľské rozhranie a počítanie rovníc. Komponenty nad rámec zadania sú dátový model a widget.

3.1 Zadanie

Úlohou alebo cieľom tejto bakalárskej práce je vývoj Android aplikácie, ktorej hlavná úloha je vyhodnocovanie užívateľom definovaných jednoduchých matematických rovníc. Ako ďalšiu funkcionalitu som sa rozhodol pridať ukladanie, upravovanie rovníc a podporu tvorby widget-ov z uložených rovníc pre rýchlejšie vyhodnocovanie bez nutnosti spúšťania celej aplikácie. Zo zadania teda vyplývajú štyri základné časti aplikácie a to užívateľské rozhranie, model pre ukladanie a prácu s uloženými rovnicami, widget a samotné počítanie rovníc.

3.2 UI

V kapitole 2.4 sme si popísali všetky princípy, ktoré by sme mali dodržiavať, aby naša aplikácia podporovala čo najviac zariadení a tým pádom bola prístupná čo najväčšiemu počtu potencionálnych užívateľov.

So všetkými podporovanými funkciami v mysli som sa rozhodol navrhnúť šesť aktivít (obrazoviek) aplikácie a to:

- Úvodnú obrazovku
- Zoznam uložených rovníc
- Pridávanie nových rovníc
- Upravovanie vybranej uloženej rovnice
- Vyhodnocovanie rovnice
- Pomoc

3.3 Model

Aby si užívateľ mohol uložiť vytvorené rovnice a nemusel ich zadávať vždy znova pri každom spustení aplikácie musíme navrhnúť ukladací systém pre pretrvávajúce dáta aplikácie.

Android podporuje viacero druhov ukladania:

Zdieľané nastavenia (Shared Preferences)
Ukladanie primitívnych dát ako páry kľúč-hodnota.

Interný úložný priestor (Internal Storage)
Ukladanie súkromných dát v pamäti zariadenia.

Externý úložný priestor (External Storage)
Ukladanie verejných dát na zdieľanom externom úložisku.

SQLite Databáze
Ukladanie štrukturovaných dát v súkromnej databáze.

Internetové pripojenie
Ukladanie dát na internet prostredníctvom vlastného sieťového serveru.

Každá z týchto možností je vhodná na niečo iné a ja som si pre ukladanie rovníc zvolil zdieľané nastavenia z dôvodu jednoduchosti ukladania a načítania rovníc prostredníctvom párov kľúč-hodnota.

3.4 Widget

Widget má poskytovať možnosť okamžitého prístupu k vyhodnocovaniu vybraných rovníc priamo z plochy telefónu, bez nutnosti spúšťať a prechádzať celú aplikáciu za účelom nájsť a vyhodnotiť vybranú rovnicu.

3.5 Počítanie rovníc

Rovnicu bude zadávať užívateľ do poskytnutého textového okna prostredníctvom aktuálne nastavenej softvérovej klávesnice zariadenia. K dispozícií bude mať taktiež základné matematické funkcie a konštanty. Po úspešnom zadaní rovnice aktivita rovnicu pošle do časti programu zodpovednej za počítanie.

Počítanie rovníc si teda musí poradiť s načítaním rovnice z reťazca v infixovej notácií (viď obrázok 4.9) a následný prevod do notácie vhodnejšej pre spracovanie, kontrolou validity rovnice, zadávaných hodnôt premenných a samotným vyhodnotením načítanej rovnice spolu s hodnotami užívateľom definovaných premenných, konštánt a základnými matematickými funkciami.

Na prevod z infixovej notácie som sa rozhodol použiť upravený shunting yard algoritmus, s ktorým prevediem infixovú notáciu na prefixovú. Výpočet bude prebiehať pomocou vstavanej knižnice `java.lang.Math`.

Kapitola 4

Implementácia

4.1 GUI

Podľa návrhu potrebujeme implementovať šesť aktivít a widget, ktorý je taktiež súčasťou užívateľského rozhrania. Toto zahŕňa prechody medzi jednotlivými aktivitami, spracovanie vstupu užívateľa, odozvu pri nesprávnych vstupoch a ak všetko prebehne podľa plánu vypísanie výsledku.

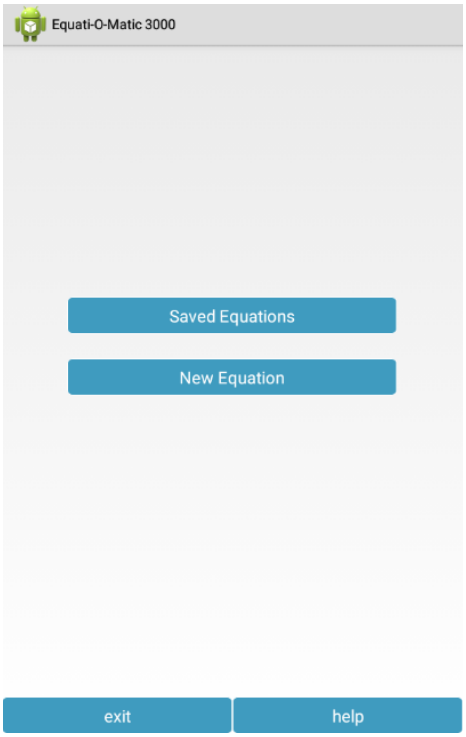
Úvodná obrazovka

Prvá aktivita (obrazovka), ktorú užívateľ po zapnutí aplikácie uvidí. Táto aktivita obsahuje tlačidlá na prechod do aktivít:

1. Uložené rovnice.
2. Vytvoriť novú rovnicu.
3. Pomoc.

Aktivita taktiež obsahuje tlačidlo na opustenie aplikácie. Podľa mnohých vývojárov by android aplikácie nemali obsahovať žiadne takéto tlačidlo(gesto, ...) a to z dôvodu fungovania Android-u. Android zabezpečuje ukončenie aplikácií sám podľa dostupných zdrojov zariadenia, na ktorom aplikácie bežia. Aplikácia na popredí má najvyššiu prioritu a Android musí zabezpečiť, aby mala všetky potrebné zdroje na jej plynulý beh. To znamená, že všetky aplikácie, ktoré momentálne niesu na popredí, sú uložené v pamäti(z dôvodu rýchlej obnovy ak by ich užívateľ znovu potreboval) a môžu byť kedykoľvek ukončené z dôvodu uvoľnenia zdrojov pre aplikáciu v popredí.

Aj napriek týmto dôvodom som sa rozhodol toto tlačidlo implementovať z dôvodu spokojnosti užívateľov. Toto tlačidlo však v skutočnosti neukončí aplikáciu, ale presunie ju do pozadia (viď obrázok 4.1).



Obrázek 4.1: Úvodná obrazovka.

Uložené rovnice

Hlavným prvkom tejto aktivity je zoznam uložených rovníc. Okrem neho aktivita obsahuje tlačidlá na:

1. Prechod do aktivity, v ktorej sa vytvárajú rovnice.
2. Vrátenie sa do predchádzajúcej aktivity.
3. Mazanie označených položiek listu(toto tlačidlo je vypnuté dovtedy, dokedy užívateľ nevyberie jednu alebo viac rovníc).

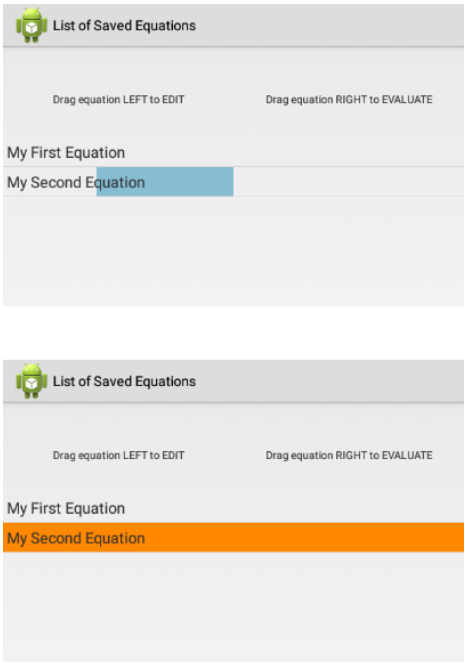
Zoznam (list) uložených rovníc je interaktívny, čo znamená, že sa s ním dajú vykonávať určité akcie.

Aktivita má tri stavy v ktorých sa môže nachádzať v závislosti na tom, ako s ňou užívateľ pracuje:

- Východzí stav (mód)
- Selekčný stav (mód)
- Ťahací stav (mód)

Po otvorení je aktivita vo východzom stave. Pri podržaní prstu na niektorej z položiek listu aktivita prejde do *selekčného módu* a vybraná položka sa označí. V selekčnom móde každé ďalšie kliknutie na neoznačené položky listu spôsobí ich označenie. Pri kliknutí na už označenú položku sa daná položka odznačí. Pri od-značení poslednej označenej položky aktivita odíde zo selekčného módu a vráti sa späť do východzieho stavu. Tlačidlo na mazanie je povolené iba počas doby keď je označená aspoň jedna položka listu.

Obrázek 4.2: Uložené rovnice.



Obrázek 4.3: Interakcia s uloženými rovnicami. Modrým je znázornený ťah, oranžovým označovanie.

Aktivita taktiež zachytáva pohyb po zozname rovníc. Ak bude užívateľ chcieť upravovať niektorú z rovníc v zozname, stačí ak potiahne po vybranej rovnici smerom doľava. Ak bude chcieť rovnicu vyhodnotiť, potiahne smerom doprava. Po prejdení dostatočnej vzdialenosti sa aktivita prepne do ťahacieho módu a ostane v ňom do doby pokiaľ užívateľ nedokončí ťahanie a prepne aplikáciu na jednu z požadovaných aktivít alebo predčasne preruší ťahanie a aktivita sa vráti do východzieho stavu.

Táto funkcionalita je implementovaná s použitím `OnTouchListener`-ov (sledovačov). Každá položka listu (rovnica) má na sebe zaregistrovaný sledovač, ktorý upozorní aktivitu v momente, keď sa prst užívateľa dotkne niektorej rovnice (položky listu). Aktivita po obdržaní tohto upozornenia spustí časovač a popritom sleduje, či užívateľ nehýbe prstom. Ak užívateľ pohne prstom o určitú vzdialenosť, časovač sa zastaví a aktivita sa prepne do ťahacieho módu. Ak užívateľ preruší dotyk, aktivita ostane vo východzom stave. Po uplynutí časovača sa aktivita prepne do selekčného módu a položka listu, ktorej sledovač poslal upozornenie bude označená. Sledovač je taktiež zaregistrovaný nad celým listom rovníc z dôvodu väčšej voľnosti pri ťahaní. Pri ťahaní je dôležitý štartovací bod ťahu, požadovaná rovnica. Samotné ťahanie už potom môže prebiehať na celom priestore aktivity (obrazovky) (viď obrázok 4.2).

Pridávanie nových rovníc, Upravovanie rovnice

Miesto(aktivita), kde je užívateľ schopný tvorby vlastných rovníc s vlastnými premennými a poskytnutými základnými matematickými funkciami a konštantami. Aktivita obsahuje tri tlačidlá na:

1. Vrátenie sa do predchádzajúcej aktivity.

2. Uloženie vytvorenej rovnice.

3. Vyhodnotenie vytvorenej rovnice (rovnica nebude uložená).

Hlavnou časťou tejto aktivity sú dve textové polia. Prvé je určené pre názov rovnice a druhé pre samotnú rovnicu. Názov sa môže skladať z ľubovoľných znakov. Ak chce užívateľ rovnicu iba vyhodnotiť, názov rovnice nieje nutné zadávať, no uloženie bez názvu nieje povolené. Rovnica musí dodržiavať určité pravidlá:

1. Každý operátor musí mať operand (číslo, premenná, funkcia, konštanta) na ľavej a pravej strane.

2. Každá funkcia musí obsahovať operand v pravo od nej (môže, ale nemusí byť v zátvorkách).

3. Všetky zátvorky musia byť uzatvorené.

4. Čísla nesmú obsahovať iné znaky než číslice.

5. Premenné nesmú obsahovať iné znaky než písmená.

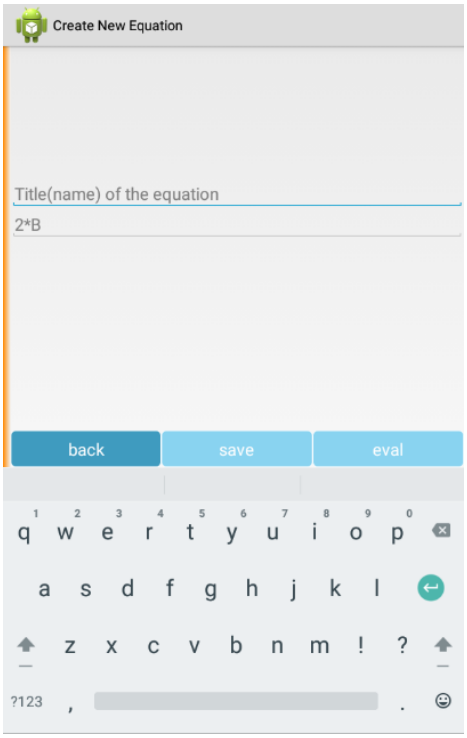
Kontrola sa vykonáva dynamicky počas zadávania rovnice užívateľom. Pri nesplnení niektorého z pravidiel je užívateľ upozornený hlásením pod rovnicou a tlačidlá pre uloženie, vyhodnotenie rovnice sú zablokované do doby než bude rovnica znova spĺňať všetky pravidlá.

Pri zadávaní rovnice je možné použiť základné matematické funkcie a konštanty, ktoré sa vyberajú z bočného menu. Toto menu sa otvára gestom „potiahnutie vpravo“(jeden krát pre funkcie, dva krát pre konštanty) a zatvára gestom „potiahnutie vľavo“. Gestá som implementoval s použitím vlastného vyhodnocovacieho algoritmu, ktorý je možné v budúcnosti použiť pre podporu zložitejších gest v aplikácií. Tento algoritmus sa skladá z:

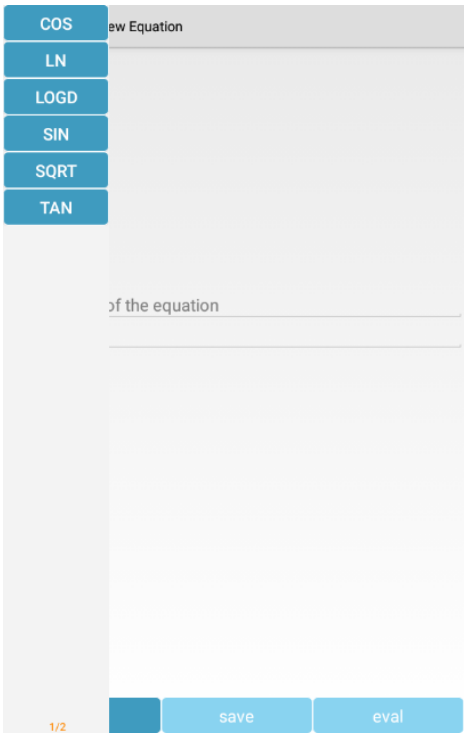
- Triedy popisujúcej vlastné rozloženie rozšírené o zachytávanie pohybu
- Triedy na vyhodnocovanie zaznamenaných pohybov
- Triedy popisujúcej gesto
- Rozhrania sledovača na hlásenie rozoznaných gest

Vlastné rozloženie sa použije v XML súbore aktivity, v ktorej chceme spracovávať gestá. Táto aktivita si potom inštanciuje triedu popisujúcu gesto s bodmi ako má gesto vyzerať. Nakoniec v aktivite vytvoríme inštanciu triedy na vyhodnocovanie, nastavíme jej vytvorené gesto/á, sledovač splnenia gesta, a celý tento objekt uložíme do triedy vlastného rozloženia vytvoreného na začiatku.

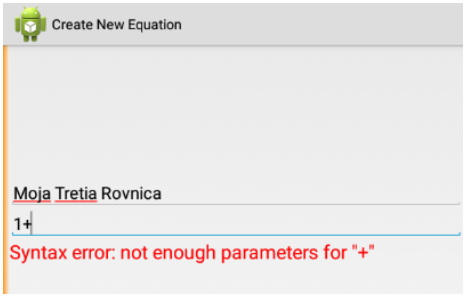
Jediný rozdiel medzi aktivitou na pridávanie rovníc a upravovanie rovníc je, že aktivita na upravovanie rovníc má namiesto tlačidla pre vyhodnotenie, tlačidlo na zmazanie upravovanej rovnice (viď obrázok 4.4).



Obrázek 4.4: Obrazovka(aktivita) pre pridávanie nových rovníc



Obrázek 4.5: Vytiahnuté menu s matematickými funkciami. Pri opätovnom potiahnutí by ho nahradilo menu s konštantami.



Obrázek 4.6: Príklad chybového hlásenia pri porušení jedného z pravidiel zadávania rovníc.

Vyhodnocovanie rovnice

Táto obrazovka(aktivita) slúži na vyhodnotenie požadovanej rovnice. Aktivita obsahuje dve tlačidlá na:

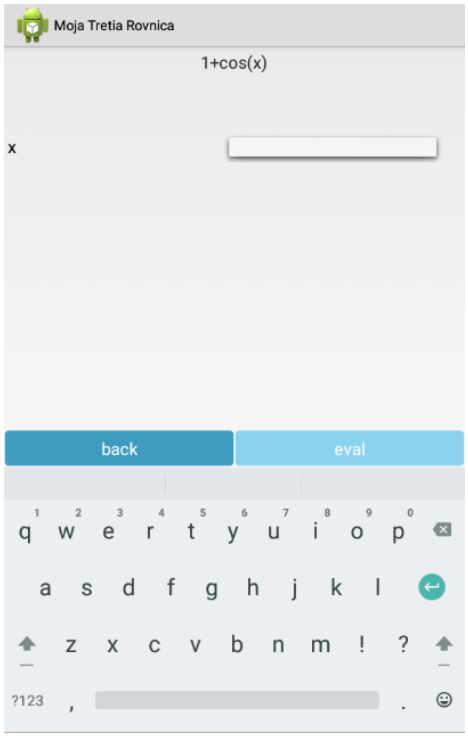
1. Vrátenie sa do predchádzajúcej aktivity.
2. Vyhodnotenie rovnice.

Ak užívateľ vytvoril rovnicu, ktorá neobsahuje žiadne premenné, aktivita zobrazí názov rovnice (ak bol zadaný) a samotnú rovnicu. Po stlačení tlačidla pre vyhodnotenie, aktivita zobrazí výsledok rovnice.

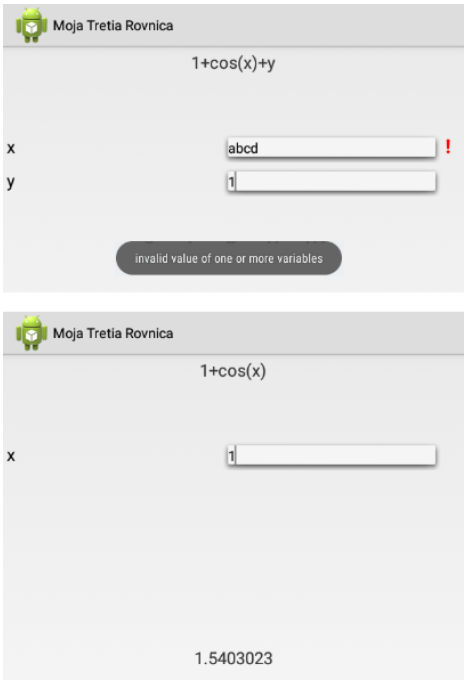
Ak rovnica obsahuje premenné, tak budú zobrazené pod rovnicou vo forme posúvateľného listu. Tlačidlo pre vyhodnotenie rovnice bude zablokované do doby, kým nebudú zadané správne hodnoty všetkých premenných. Za správnu hodnotu sa považuje iba číselná hodnota. Kontrola zadávanej premennej sa vykonáva dynamicky, a v prípade chyby je užívateľ informovaný červeným výkričníkom vedľa zadávanej hodnoty a chybovým hlásením v spodnej časti displeja. Prázdna hodnota premennej nieje považovaná za chybu(nieje zobrazený výkričník ani zablokované vyhodnotenie), no po stlačení tlačidla pre vyhodnotenie je užívateľ upozornený na ne-inicializované premenné chybovým hlásením (viď obrázok 4.7).

Pomoc

Aktivita s jedným tlačidlom na vrátenie sa k prechádzajúcej aktivite a základnými inštrukciami ako ovládať aplikáciu.



Obrázek 4.7: Obrazovka(aktivita) pre vyhodnocovanie rovnice



Obrázek 4.8: Interakcia pri vyhodnocovaní rovnice.

Widget

Pri tvorbe widget-u, si užívateľ vyberie požadovanú rovnicu zo zoznamu uložených rovníc. Následne sa mu na pozadí vytvorí widget, ktorého veľkosť sa dá meniť, a ktorý obsahuje názov rovnice, ktorú si užívateľ vybral. Widget je čiernobiely s bielym textom rovnice, okrajom a čiernym pozadím.

Po kliknutí na widget konkrétnej rovnice je užívateľovi zobrazené aktivita pre vyhodnocovanie, kde je zobrazená rovnica identifikovaná názvom špecifikovaným v kliknutom widget-e.

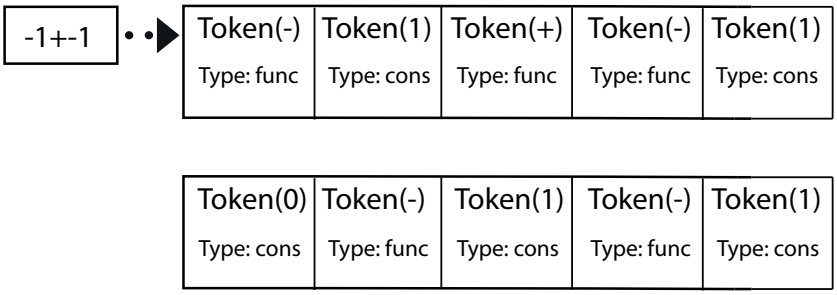
4.2 Implementácia algoritmu na vyhodnocovanie rovníc

Vstup do výpočtovej časti programu je rovnica v tvare reťazca. Tento reťazec musí byť prevedený do tvaru vhodného pre výpočet. Najprv je rozdelený do listu *tokenov*, ktorý je následne potrebné previesť z *infixovej* notácie (viď obrázok 4.9), ktorá je nepoužiteľná pre výpočet. Rozhodol som sa pre *poľskú notáciu (prefixovú)*, z dôvodu následného, pre mňa jednoduchšieho, vyhodnocovania. Tento reťazec už v poľskej notácií bolo potrebné previesť do vnútornej reprezentácie programu. Každá časť rovnice, konštanta, premenná alebo funkcia (operátor je špeciálny typ funkcie), je implementovaná príslušným objektom programu. Každý objekt obsahuje metódu na vyhodnotenie, ktorá je volaná nad každou časťou rovnice.

Lexikálna analýza

Spracováva rovnicu vo forme reťazca. Po znaku vytvára jednotlivé tokeny, ktoré sú ukladané do zoznamu tokenov [15]. Každý token v sebe nesie načítanú hodnotu spolu s jej typom (konštanta, funkcia, premenná, zátvorky). Tento list sa pred predaním ďalším častiam programu ešte raz prejde, aby sa vyhodnotili znamienka plus a mínus. Táto časť je dôležitá z dôvodu, že každý operátor potrebuje dva operandy. Znamienka môžu byť viacnásobné alebo vo forme vyjadrenia zápornej či kladnej hodnoty čísla. Pred unárne operátory vyjadrujúce zápornú, alebo kladnú hodnotu sa vloží konštanta s hodnotou 0. Viacnásobné znamienka sa nahradia výsledným znamienkom (viď obrázok 4.10).

Obrázek 4.9: Infixová, prefixová a postfixová notácia [23].



Obrázek 4.10: Grafické znázornenie lexikálnej analýzy.

Shunting Yard

Výstupom je *postfixová* notácia. Shunting Yard je zásobníkovo orientovaný algoritmus. Na prevod sa okrem zásobníka používajú ešte dve ďalšie premenné na vstup a výstup. Zásobník sa používa na ukladanie operátorov, ktoré ešte neboli pridané do výstupu.

Aby sme mohli analyzovať časovú náročnosť algoritmu, musíme si najprv uvedomiť, že každý token sa prečíta raz. Každé číslo, funkcia alebo operátor sa vypíšu raz. Každá funkcia, operátor alebo zátvorka sa pridajú a odoberú zo zásobníka raz. V najhoršom prípade sa preto vykoná konštantný počet operácií na token, čo znamená, že doba vykonávania programu je $O(n)$.

Prevod postfixovej notácie do prefixovej

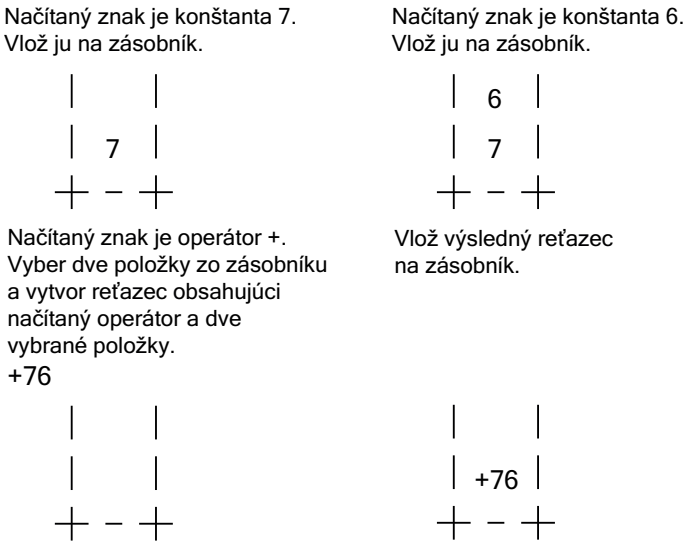
Je možné vyhodnocovať priamo postfixovú notáciu, no ja som sa rozhodol, v rámci zjednodušenia vyhodnocovania, previesť postfixovú notáciu do prefixovej. Prevod sa deje pomocou zásobníku. Načítané konštanty sú ukladané na zásobník. Ak sa prečíta operátor alebo funkcia, tak je príslušný počet položiek vybraný zo zásobníka a vložený spolu s operátorom alebo funkciou do nového listu, ktorý je znova uložený na zásobník (viď obrázok 4.11). Po spracovaní všetkých tokenov, ostáva na zásobníku jedna položka, v ktorej je uložený list tokenov v prefixovej notácií [11].

Prevod tokenov do formy vhodnej na vyhodnotenie

Tokeny v zozname sú podľa typov nastavených v lexikálnej analýze prevedené na príslušné objekty. Pre token typu konštanta a premenná je vytvorená nová inštancia prislúchajúcej triedy. Konštanty obsahujú iba hodnotu, zatiaľ čo premenné obsahujú názov premennej a odkaz do tabuľky premenných, kde bude uložená užívateľom zadaná hodnota pri vyhodnotení rovnice. Pre tokeny typu funkcia (operátor je špeciálna funkcia s 2 parametrami) sa kopírujú príslušné inštancie z tabuľky funkcií. Tabuľka obsahuje všetky podporované funkcie, ktoré sú vytvárané pri spustení aplikácie. Funkcie obsahujú názov (napr. $+$) a metódu na vyhodnotenie. Tieto kópie nemajú naplnené operandy potrebné pre konečný výpočet. Vo výsledku teda dostaneme z listu tokenov, list objektov.

Vytvorenie konečného výrazu

Na prvý objekt uložený v zozname sa zavolá metóda na jeho vrátenie s naplnenými operandmi. Pre konštantu a premennú to znamená vrátenie ich hodnoty. Funkcia najprv musí

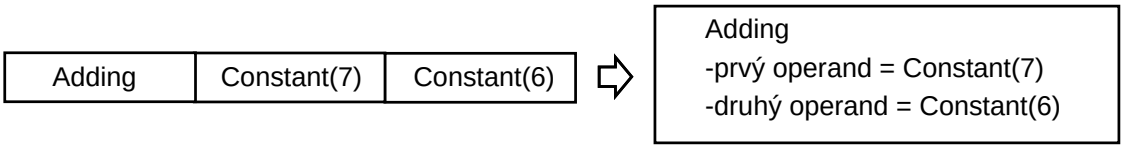


Obrázek 4.11: Grafické znázornenie prevodu postfixovej notácie na prefixovú. Príklad je pre vstup 76+.

zavolať tú istú metódu nad svojimi operandmi. Toto nám vytvára rekurziu, ktorej výsledkom je jeden výraz s nastavenými operandmi pripravený na vyhodnotenie (viď obrázok 4.12). Toto je riešenie, ktoré malo za účel zvýšiť rýchlosť pri opätovnom vyhodnocovaní, kde užívateľ môže zmeniť hodnoty premenných v rovnici, no tvar samotnej rovnice sa už nezmení. Výraz sa teda pri vyhodnocovaní vytvorí iba raz, čo vedie k rýchlejšiemu následnému vyhodnocovaniu.

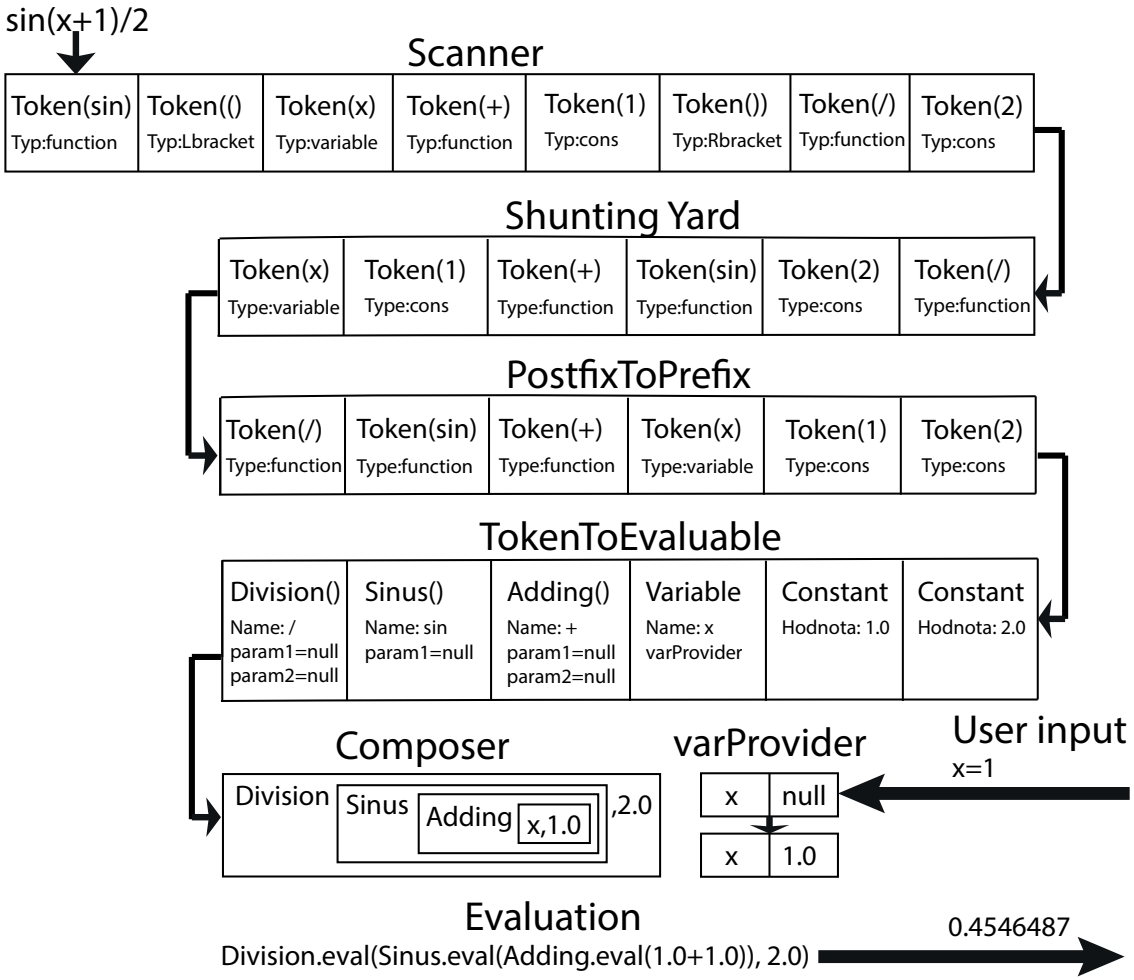
Vyhodnotenie rovnice

Každý objekt rovnice (funkcia, konštanta, premenná) obsahuje metódu na vyhodnotenie. Pri funkciách to znamená vrátenie výsledku operácie s operandmi, konštanty a premenné vracajú svoju hodnotu. Táto metóda sa zavolá rekurzívne na konečný výraz a v ňom na jeho operandy. Výsledkom metódy bude číslo typu `float`.



Obrázek 4.12: Grafické znázornenie skladania výsledného výrazu, ktorý bude vyhodnotený.

Príklad vyhodnotenia rovnice



Obrázek 4.13: Grafické znázornenie výpočtu rovnice od momentu zadania rovnice až po získanie výsledku.

Kapitola 5

Záver

Hlavným cieľom tejto bakalárskej práce bol vývoj editora matematických rovníc pre Android. Podľa môjho názoru som tento cieľ naplnil. Aplikácia zvláda počítanie rovníc s matematickými funkciami, konštantami a premennými.

Túto funkcionalitu majú mnohé existujúce aplikácie, ktoré môžeme nájsť v Google Play obchode. Rozdiel medzi týmito aplikáciami a mojou je, že užívateľ si môže definovať **vlastné** rovnice s **vlastnými** premennými.

Súčasťou mojej aplikácie je taktiež widget, ktorý slúži na rýchle vyhodnocovanie jednotlivých uložených rovníc bez nutnosti prechádzania celej aplikácie. Tento widget som nenašiel u žiadnej existujúcej aplikácie určenej na prácu s rovnicami.

Vyhodnocovanie rovníc (výrazov) zo sebou prináša určité problémy, ako napríklad rovnice vo forme čitateľnej pre ľudí (infixová notácia), ktoré sú nevhodné pre spracovanie počítačmi.

Pri vývoji užívateľského rozhrania aplikácie som sa snažil uplatniť správne princípy dizajnu a moderný prístup pri interakcii s užívateľom. Príkladom môže byť použitie gest pri otváraní menu alebo sledovanie ťahania pre upravovanie, vyhodnocovanie rovnice.

Pri ďalšom vývoji aplikácie by som sa mohol inšpirovať existujúcimi aplikáciami popísanými v teoretickej kapitole tejto technickej správy. Ako príklad môžem uviesť zdieľanie rovníc medzi priateľmi, vlastnú klávesnicu pre zadávanie rovníc alebo export a import rovníc z textového súboru.

Pri písaní tejto bakalárskej práce som sa naučil ako vyvíjať mobilné aplikácie pre platformu Android OS, akú dôležitú rolu zohráva v aplikácií užívateľské rozhranie a rozšíril som si znalosti objektovo orientovaného programovania v programovacom jazyku Java.

Literatura

[1] Designing for Multiple Screens. Android Developer.
URL http://developer.android.com/training/multiscreen/index.html

[2] Layouts. Android Developer.
URL https://developer.android.com/guide/topics/ui/declaring\
discretionary{-}{}{}layout.html

[3] UI Overview. Android Developer.
URL https://developer.android.com/guide/topics/ui/overview.html#Layout

[4] Widgets. Android Developer.
URL http://developer.android.com/design/patterns/widgets.html

[5] Equation solving. 2015-03-21.
URL http://en.wikipedia.org/wiki/Equation_solving

[6] Aho, A.; Lam, M.; Ullman, J.; aj.: *Compilers: Principles, Techniques, and Tools*. Pearson Education, 2011, ISBN 9780133002140.

[7] Apache: Apache Software License, Version 2.0. 2004-1.
URL http://www.apache.org/licenses/LICENSE-2.0

[8] Bendersky, E.: Top-Down operator precedence parsing. 2010-1-02.
URL http:
//eli.thegreenplace.net/2010/01/02/top-down-operator-precedence-parsing

[9] Bendersky, E.: Parsing expressions by precedence climbing. 2012-8-02.
URL http://eli.thegreenplace.net/2012/08/02/
parsing-expressions-by-precedence-climbing#id5

[10] Burov, S.: Programmable Calculator. 2012-10-14.
URL https://play.google.com/store/apps/details?id=ru.stanislavburov.
android.PGMCalc&hl=en

[11] CProgrammer: Postfix To Prefix Conversion.
URL http:
//see-programming.blogspot.in/2013/05/postfix-to-prefix-conversion.html

[12] Djikstra, E.: Supplement nr. 10. The University of Texas at Austin, 1961.
URL http://www.cs.utexas.edu/~EWD/MCReps/MR35.PDF

[13] Doyle, P. M.: Science Equation Solver. 2013-8-15.
URL
https://play.google.com/store/apps/details?id=your.science.solver&hl=en

[14] GranadaWare: Equation Editor. 2013-7-8.
URL https:
//play.google.com/store/apps/details?id=com.vic.equationeditor&hl=en

[15] Gries, D.: *Compiler Consruction for Digital Computers.* Wiley, 1971.

[16] Holetzeck, J.: Kalkulon. 2013-6-8.
URL
https://play.google.com/store/apps/details?id=de.kalkulon.main&hl=en

[17] Jackson, W.: *Android Apps for Absolute Beginners.* For Absolute Beginners, Apress, 2012, ISBN 9781430247883.

[18] Katsukawa, A.: Advanced Equation Solver. 2014-9-5.
URL https:
//play.google.com/store/apps/details?id=org.advancedequationsolver&hl=en

[19] Malik, S.: Math Solver. 2015-3-11.
URL https:
//play.google.com/store/apps/details?id=com.shakti.mathssolver&hl=en

[20] Mawston, N.: Android Hits Record 85 Percent Share of Global Smartphone Shipments in Q2 2014. Strategy Analytics, 2014-07-30.
URL https://goo.gl/fNJ5mt

[21] Norvell, T.: Parsing Expressions by Recursive Descent. Memorial University, 2006-9-14.
URL http://www.engr.mun.ca/~theo/Misc/exp_parsing.htm

[22] QxMD: Calculate by QxMD. 2014-10-9.
URL
https://play.google.com/store/apps/details?id=com.qxmd.calculate&hl=en

[23] Rawat, K.: Infix, prefix and postfix notations (Polish Notations). 2012.
URL
http://www.ritambhara.in/infix-prefix-postfix-notations-polish-notations

[24] Vogel, L.: Android application and activity life cycle - Tutorial. 2015-02-09.
URL http://www.vogella.com/tutorials/AndroidLifeCycle/article.html

Příloha A

Obsah CD

- `/app` - zdrojové súbory android aplikácie
- `/apk` - skompilovaná aplikácia
- `/tz` - táto technická správa
- `/tz/latex` - zdrojové súbory technickej správy
- `/demo` - demonštračné video